03513844
SOFTWARE *TESTING* SYSTEM

>PUB. NO.: 03-176744 [JP 3176744 A]
PUBLISHED: July 31, 1991 (19910731)
INVENTOR(s): SASAKI MICHITAKA
KOIZUMI AKINORI
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP (Japan)
HITACHI COMPUT ENG CORP LTD [472484] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 01-315310 [JP 89315310]
FILED: December 06, 1989 (19891206)

ABSTRACT

PURPOSE: To reduce the manhour of command correction by automatically correcting a command for instructing debugging in a file in accordance with program *updating* at the time of *updating* the program in a test debugging system.

CONSTITUTION: An old source program 14a is *updated* by an editor 11 to a new source program 14b. In the case of *updating* the source program 14a, an editor 11 outputs the line number between an adding line and a deleting line as a difference line information 15. A debugging *updating* program 12 refers the information 15, corrects an old debugging command 16a correspondingly to the new source program 14b and outputs a new debugging command 16b.

⑲日本国　許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平3−176744

⑤Int.Cl.⁵　識別記号　庁内整理番号　⑬公開　平成3年(1991)7月31日
G 06 F 11/28　A　8522−5B

審査請求　未請求　請求項の数 1　(全5頁)

⑭発明の名称　ソフトウエアテスト方式

⑳特　願　平1−315310
㉒出　願　平1(1989)12月6日

㉑発明者　佐々木　道孝　神奈川県秦野市堀山下1番地　日立コンピュータエンジニアリング株式会社内
㉑発明者　小泉　昭典　神奈川県秦野市堀山下1番地　日立コンピュータエンジニアリング株式会社内
㉑出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地
㉑出願人　日立コンピュータエンジニアリング株式会社　神奈川県秦野市堀山下1番地
㉔代理人　弁理士　小川　勝男　外1名

明　細　書

1．発明の名称

ソフトウェアテスト方式

2．特許請求の範囲

1．プログラムのテスト及びデバッグを指示するコマンドをファイルから入力して実行することができるテスト・デバッグシステムに於いて、ソースプログラムの更新時に変更内容を示す更新情報を出力できるソースプログラム編集部と、該更新情報に従って該コマンドを修正できるコマンド修正部から成り、ソースプログラムの更新時に、該コマンドを該ソースプログラムの変更内容に合せて自動的に修正することを特徴とするソフトウェアテスト方式。

3．発明の詳細な説明

〔産業上の利用分野〕

本発明は、ソフトウェアのテスト方式に係り、特に、プログラムのテスト及びデバッグを指示するコマンドをファイルから入力することができるテスト・デバッグシステムに於いて、ファイル中に保存されている該コマンドを効率よく修正する方法に関する。

〔従来の技術〕

従来の方式は、システム使用者が文番号やラベル名を指定してコマンドを作成し、テスト・デバッグを行うものであり、ソースプログラムとそれに対応するコマンドは、それぞれ独立に更新・修正する方式である。

〔発明が解決しようとする課題〕

従来技術では、ソースプログラム更新時に変化する行番号や文番号などのプログラムの位置情報を、コマンド中に使用している場合、コマンドの修正し易さについて配慮がされておらず、プログラム更新後にファイル中の該コマンドを見直し、人手で、更新後のプログラムに合わせて修正しなければならないという問題点があった。本発明は、この問題点を解決するためになされたものである。

本発明の目的は、ソースプログラムを更新した時に、該ソースプログラムの更新内容に合わせて、ファイル中の該コマンドを自動的に修正すること

によって、コマンド　正工数を削減することにある。

〔課題を解決するた　手段〕

上記目的を達成するために、本発明においては、ソースプログラムを更新する際に、変更内容を表す更新情報を出力し、該更新情報を参照することによって、ファイル中のコマンドで使用しているソースプログラム更新時に変化する行番号や文番号などのプログラムの位置情報を、ソースプログラムの変更内容に合わせて自動的に修正するようにしたものである。

〔作用〕

ソースプログラムを更新する際は、行単位に追加、削除、変更が行なわれる。この時、追加行番号、削除行番号を更新情報として出力しておく。コマンドをファイルから入力して実行することができるテスト・デバッグシステムで更新後のプログラムをテスト・デバッグする前に、該更新情報を参照して、ファイル中の該コマンドで使っているソースプログラム更新時に変化する行番号や文番号などのプログラムの位置情報を、変更内容に合わせて修正する。これによって、プログラムとファイル中の該コマンドの指示が不一致になるということがなくなる。

〔実施例〕

以下、本発明の実施例2つを図面を用いて具体的に説明する。

第1図、第2図及び第3図は、ソースプログラムを入力し、プログラムのテスト・デバッグを行う装置のデバッガに、本発明を適用した実施例を説明するための図面である。

第1図は、本発明を実施したテスト・デバッグシステムのデータフロー図である。11は、ソースプログラムを更新する際に使用するエディタ、13は、ソースプログラム及びファイル中のデバッグコマンドを入力しテスト・デバッグを行うデバッガである。14aは、エディタ11で更新する前の旧ソースプログラムであり、行番号は一定間隔で昇順に振られているものとする。14bはエディタ11で更新した後の新ソースプログラムである。16aは、旧ソースプログラム14aをテスト・デバッグするためにファイル中に作成した旧デバッグコマンド、16bは、プログラムの更新内容に応じて修正した、新ソースプログラム14bをテスト・デバッグするための新デバッグコマンドである。

エディタ11は、ソースプログラムを更新する際、追加行及び削除行の行番号を差分行情報として出力する。15は、差分行情報である。12は、差分行情報15を参照し、旧デバッグコマンド16aを、新ソースプログラム14bに対応させて修正し、新デバッグコマンド16bを出力する、デバッグコマンド更新プログラムである。

第2図は、差分行情報のレコードフォーマットである。21は、旧ソースプログラム14a上での追加行の直前の行番号であり、22は、追加行の行数である。23は、旧ソースプログラム14a上での削除行の行番号である。

第3図は、デバッグコマンド更新時の処理フローである。31はファイル中の旧デバッグコマンドの先頭から最後まで一つ一つ32以降の処理を繰り返すことを示す。32は、ファイル中の旧デバッグコマンドのうち、コマンド中に行番号を使っているもののみ33以降の処理を行うことを示す。33は、差分行情報の追加行レコードの先頭から最後まで、一つ一つに、34及び35の処理を行うことを示す。34では、デバッグコマンド中の行番号と追加行レコードの追加行番号を比較し、デバッグコマンド中の行番号の方が大きかったら35の処理を行うことを示す。35は、デバッグコマンド中の行番号に追加行レコード中の追加行数分の行番号を加算することを示す。

36は、差分行情報の削除行レコードの先頭から最後まで、一つ一つに、37，38及び39の処理を行うことを示す。37は、デバッグコマンド中の行番号と差分行情報の削除レコード中　削除行番号を比較し、デバッグコマンド中　行番号の方が大きかったら38の処理を行い、等しかったら39の処理を行うことを示す。38は、デバッグコマンド中の行番号から一行分の行番号を減算することを示す。39は、デバッグコマンドが無効になったので削除

することを示す。

デバッグコマンドの更新処理が終った後、新ソースプログラムの行番号を旧ソースプログラム14aと同じ間隔で昇順に振り直す処理を行う。これによって、デバッグコマンド更新によって更新された新デバッグコマンド中の行番号は、新ソースプログラムの行番号と対応が取れる。

第4図及び第5図は、ソースプログラムをコンパイルして得たオブジェクトプログラムを入力し、プログラムのテスト・デバッグを行う種類のデバッガに、本発明を適用した実施例を説明するための図面である。

第4図は、本発明を実施したテスト・デバッグシステムのデータフロー図である。11は、第1図中の11と同じエディタであり、旧ソースプログラム14aを更新する際に使用し、新ソースプログラム14bを出力する。このとき、第2図と同じレコードフォーマットの差分行情報15を出力する。

41はコンパイラであり、ソースプログラムを入力し、オブジェクトプログラム45を出力する。このとき、入力したソースの行番号とコンパイル後の文番号の対応を示す行・文対応情報46bを出力する。46aと46bは、取方共、行・文の対応情報であるが、46bは、更新後の新ソースプログラム14bをコンパイルしたときに出力される新行・文の対応情報であり、46aは、更新前の旧ソースプログラム14aをコンパイルしたときに出力された旧行・文の対応情報である。

42は、差分文情報作成プログラムであり、新行・文対応情報46bと旧行・文対応情報46aを参照し、差分行情報15から、追加文の位置と文数及び削除文の位置を示す差分文情報47を作成する。

43は、デバッグコマンド更新プログラムであり、差分文情報47を参照し、旧デバッグコマンド48aを、プログラムの更新内容に対応させて修正し、新ソースプログラム14bをコンパイルして作成したプログラムに対して、旧デバッグコマンド48aと同等のテスト・デバッグを行う、新デバッグコマンド48bを出力する。

44は、デバッガであり、更新後の新ソースプログラム14bをコンパイルして作成したオブジェクトプログラムを、デバッグコマンド更新プログラム43で更新した新デバッグコマンド48bを使って、テスト・デバッグを行う。

なお、新ソースプログラム14bの行番号の振り直しは、デバッグコマンド更新プログラム43でデバッグコマンドを更新した後に行う。そのとき、振り直した行番号に対応して、新行・文の対応情報46bの行番号を更新し、次回のソースプログラム更新に備える。

第5図は、差分文情報のレコードフォーマットである。51は、旧プログラム上の追加文の直前の文番号であり、52は追加文の文の数である。53は、旧プログラム上の削除文の文番号である。なお、差分文情報のレコードフォーマットは、行が文に変るだけで、第2図の差分行情報と同一である。

第4図中のデバッグコマンド更新プログラム43がデバッグコマンドを更新する際の処理フローは、行が文に変るだけで、第3図の処理フローと同一である。

（発明の効果）

本発明によれば、プログラムのテスト及びデバッグを指示するコマンドをファイルから入力して実行することができるテスト・デバッグシステムにおいて、プログラムを更新したとき、更新内容に対応させて、ファイル中の該コマンドを自動的に修正する。これによって、更新前のプログラムに対して作成したコマンドと同等のテスト及びデバッグを、更新後のプログラム対しても自動的に行えるようになる。

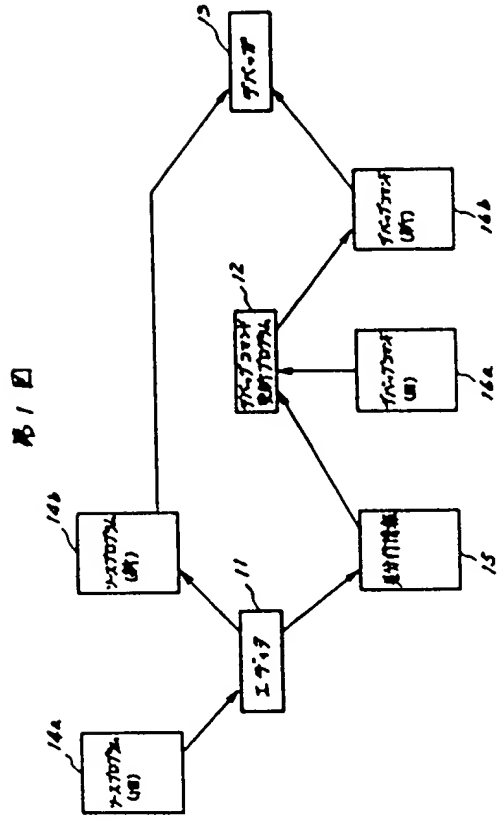
4．図面の簡単な説明

第1図はソースプログラムを入力し、プログラムをテスト・デバッグする種類のデバッガに本発明を適用した場合のテスト・デバッグシステムのデータフロー図，第2図は差分行情報のレコードフォーマットを示す図，第3図はデバッグコマンド更新時の処理フロー図，第4図はオブジェクトプログラムを入力し、プログラムをテスト・デバッグする種類のデバッガに本発明を適用した場合のテスト・デバッグシステムのデータフロー図，

第5図は差分文書号情報　レコードフォーマットを示す図である。

11…エディタ、12及び43…デバッグコマンド更新プログラム、15…差分行情報、46a及び46b…旧及び新行・文の対応情報、47…差分文情報。

第1図



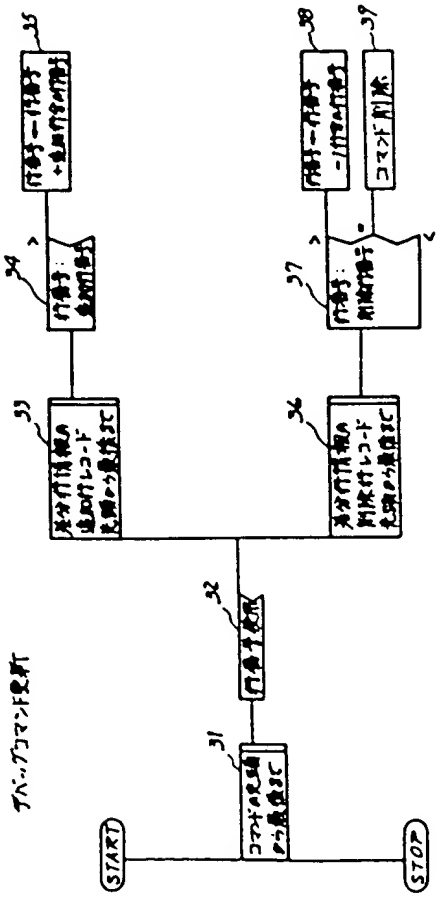
代理人弁理士　小　川　勝　男

第2図

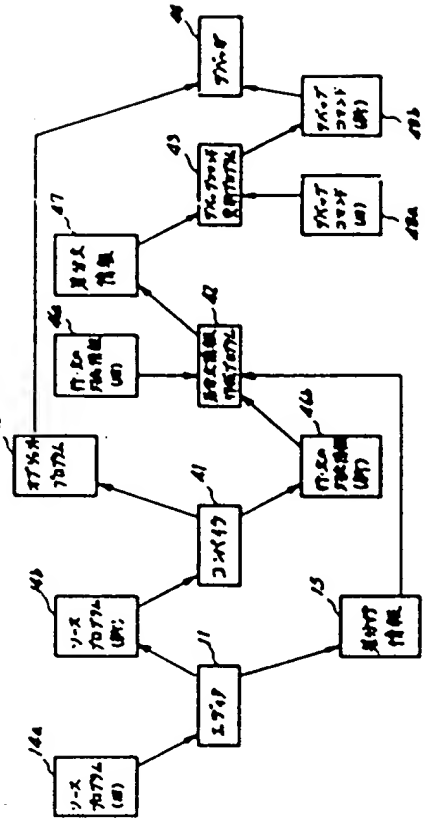
差分行情報　15

追加	直前の行番号	21
	行数	22
削除	行番号	23

第3図



第4図



第5図

差分文情報 47

追加	直前の文番号		51
	文の数		52
削除	文番号		53